International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 10 | Oct -2016 www.irjet.net p-ISSN: 2395-0072

Security Enhancement by Achieving Flatness in Honeywords

Ajab Karishma[1], Borchate Pranali[2], Jadhav Ashwini[3] Jadhav Shubhangi[4]

[1234], Dept Of Computer Engineering, SGOI COE, Marashatra, India

Abstract - Every year new approach against cyber security threats are introduced. But simultaneously the adversary also create new techniques those overcome these efforts. So considering for security and data protection as a priority new techniques are needed. So, there is one of the important security issues is with disclosure of password file. To overcome this issue we introduce honeywords (fake passwords) to detect attacks against hashed password databases. For each user account actual password is stored with honeywords. Adversary who steals a file of hashed passwords cannot be sure if it is the actual password or a honeyword for any account. If adversary enters the honeyword for login then it will trigger an alarm notifying the administrator that password file breach. In our system, if the number of attempts exceeds the count of three or entered password other than honeywords then the access will be issued but the files available will be decoy files. Also we suggest an alternative approach to provide realistic honeywords a perfectly flat honeywords generation approach and also to reduce storage cost of the honeywords scheme

Key Words: Honeywords, Honeypot, Authentication, Security, Salting, Password hashing.

1.INTRODUCTION

In authentication process it becomes difficult to handle security of passwords that's why password became the most important asset to authenticate. But users choose the passwords that are easy to remember that can be predicted by the attacker using different attacks like brute force, dictionary, rainbow table attacks etc. So Honeywords plays an important role to defence against stole password files. Specifically, fake passwords placed in the password file of an authentication server.

Generally in many software industries and companies store their data in databases like ORACLE, Mysql or may be other. So, the starting point of a system which is required user name and password are stored in database. Once a password file is bagged, by applying the password cracking technique it is easy to capture most of the plaintext passwords. So for skipping it, there are some issues that should be considered to overcome these security problems:

- First passwords must be safe and secure by using the relevant algorithm.

- Second point is that a secure system should detect the entry of unauthorised user in the system.

In this study focus is on the use of fake passwords and accounts. The administrator deliberately creates fake accounts and detects a password disclosure, if any one of the honeypot account get used it will detect by admin. According to the survey, some incorrect login attempts password for each user, the honeypot accounts of other words is known malicious behaviour.

1.1 Honeyword:

The idea is the insertion of fake passwords called as honeywords associated with each user's account. When an attacker gets the password file, he recovers many passwords for each account and he cannot be sure

about which word is genuine. Hence, the cracked password files can be detected by the system administrator if a login attempt is done with a honeyword by the adversary. Honeywords concept of Juels and Rivest is totally based on the generation of honeywords which is done by Gen( ) algorithm and this is easy to crack and find the correct password. For generating these Honeywords Juels and Rivest use some methods these are Chaffing-by-tweaking, Chaffing-with-a-password-model, Chaffing with-Tough Nuts and Hybrid Method. These methods are useful and decrease the chances of guessing correct password.

1.2 Honeypot:

A Honeypot is a security capability whose value is being probed, attacked or comprised.

A honeypot is a secure source.

A Honeypot could just as simply be one of your old PC's, a script or even a digital entity like some fabricated patient records. Whose value is being probed, attacked or comprised.

1.3 Honeyindex:

Instead of honeywords we use honeyindexes, for every account we created a new and unique honeyindex. The correct honeyindex is store with the hash of the correct password in a list. In another list we have integer list with the username, the integer list is a honeyindexes of other accounts as well as their own account honeyindex this list is called as a honeyindex set.

2. Literature survey

The most important concept is information security requirement in this which is secured using some authentication method. Various authentication method are existing such as Patterns, Passwords, PIN's etc.. Now-a-days most generally used technic for authentication is passwords. Security of password is an important part in security. A password is a secret word, which a user must input during a login, this word is match only after that it is possible to get access.

Generally disclosure of password files is a several security problem that has affected millions of users and many companies and software industries store their data in database, Like facebook, Yahoo, RockYou, Gmail and Adobe[1],[2]. Generally user name and passwords are stored in a database. Since stolen passwords make the users target of many possible attacks. These recent events have proved that the weak password storage methods are currently used by many people on websites. For example, the LinkedIn passwords were using the SHA-1 algorithm without a salt and similarly the passwords in the eHarmony system were also stored using unsalted MD5 hashes[3].Once a password file is leakage, attacker by using the password cracking technique it is easy to capture most of the plaintext passwords[4].

In this respect, there are two issues that should be considered to avoid these security problems: First, passwords must be protected by taking proper caution and storing with their hash values computed through some other correct complex mechanisms. Hence, for an advance it must be hard to include hashes value in plaintext passwords. The second point is that a secure system should detect whether a password file leakage incident happened or not to take appropriate actions. Honeypot is one of the methods to identify occurrence of a password database breach. In this approach, the administrator purposely creates deceit user accounts to lure adversaries and detects a password disclosure, if any one of the honeypot passwords get used [5], [6]. In the proposed system we focus on the honeyindex

and deal with fake passwords or accounts as a simple and cost effective solution to detect compromise of passwords.

Many researchers have already worked for password security approach. Earlier, to protect online banking accounts from brute-force attacks, Herley and Florencio [7] proposed a new approach to detect the malicious behaviour on every incorrect or unauthorized login. For every single user false login attempts with few passwords will generate honeypot accounts (fake accounts) so that malign behaviour is caught. Recently, Juels and Rivest have presented the honeyword mechanism to detect an adversary who attempts to login with cracked passwords [8]. Imran Erguler's Achieving Flatness by Selecting the Honeywords from Existing User Passwords[8],this suggest an alternative approach that selects the honeywords from existing user passwords in the system in order to provide realistic honeywords a perfectly flat honeyword generation method and also to reduce storage cost of the honeyword scheme[9]. The propose system in concept is that for each username they build a set of honeyindexes in which one is real index and the others are false. When detecting the honeyword than alarm is triggered which notifies the administrator about the password file breach.

3. Architecture

3.1 Registration

Username and password are required from the user to register the system. Honeyindexes are generated periodically honeyindex set of each account should be regenerated.

.

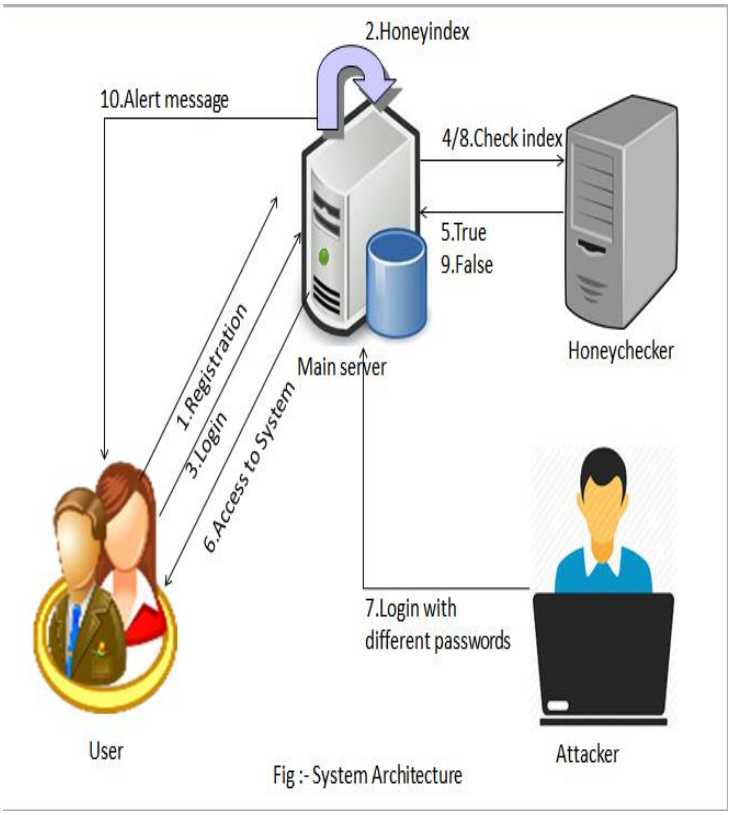

Fig -1: Architecture diagram

3.2 Honeychecker

Is an auxiliary service, honeychecker is employed to store correct indexes for each account. The honeychecker executes two commands sent by the main server:

1) Set: c_i,u_i

Sets correct password index ci for the user u_i.

2) Verified: u_i,j

Verifies whether ci for u_i is equal to given j. Returns the result and if equality does not hold, notices system a honeyword situation. Thus, the honeychecker only knows the correct index for ausername, but not the password or hash of the password.

3.3 Login

• System firstly checks whether entered password, g, is correct for the corresponding username u_i.

• Then, the hash values stored in f1 file for the respective indices in k,S_i are compared with H(g) to find a match.

• If a match is not found, then it means that g is neither the correct password, nor one of the honeywords, i.e. login fails.

• If H(g) is found in list, then the main server checks whether the account is a honeypot. If it is a honeypot, then the attacker is get redirected to the fake application and log is get created.

• If, however, H(g) is in the list and it is not a honeypot, the corresponding j S_i is delivered to honeychecker with username as <u_i,j> to verify that it is the correct index.

• Honeychecker checks whether j=c_i and returns the result to the main server. At the same time, if it is not equal, then it assured that password is a honeyword and alert is send to the main user about someone is trying to access your account.

3.4 Database

• In previous database module the new entries stored in the sequence they are inserted i.e. in the sequence new users are registered.

• To secure our database we mainly focus on shuffling of the records in the tables.

• We had Shuffled records in the tables in the two ways

• Shuffling of usernames and index value

• Shuffling of honeyindexes in honeyindex set

• We had shuffled the username so that hacker will not easily get the data of the registered user and we also shuffled index value of related user name shuffling of honeyindex set is the main part in our project. Here we shuffling the honeyindexes in the honeyindex set so if hacker will get the honeyindex set then also he will not able to find there cords from its position because the position of every honey value in honeyindex set is shuffled.

3.5 User Login

Here user is going to Login into the System. If password matches with the hash password then user can Login.

3.6 Hacker

Here hacker login to the system. Here if hacker tries to access the system and if he enters any honeyword then the notification or alert message is given to the Actual user.

3.7 Log Creation

Log creation is done for every user actions to the system and which is store into the database.

4. Proposed System

Our proposed model is still using honeywords concept. However, instead of generating honeywords we generate honeyindexes of existing passwords. For achieving this, for each account we assign index number, which we call honeyindexes. Moreover, hash of the correct password is saving with the correct index in a list. On the other side, in another list u_i is stored with a honeyindex set which is consisting of the honeyindexes and also the correct index. Honeyindex set is created as when any new user registered it takes some honeyindexes and then merge the new honeyindex of that new user and shuffled all the honeyindexes after shuffling we get the new honeyindex set which will get stored in the list. Whenever new user registered all the rows in second list get shuffled. So, when hacker analyses the two lists, he recognizes that each username is matched with k numbers as sweetindexes and each of points to real passwords in the system. The tentative password indexes box an adversary to make a precise guess

and he cannot be easily sure about which index are the correct on and other hand shuffling of records and honeyindexes in honeyindex set increases the complexity. The contribution of our approach: First, this model requires less storage compared to the original study. Second, effectiveness of the honeyword system directly depends on how Gen() flatness is creating the honeywords and how it depends on human behaviour in choosing passwords. In our approach indexes of passwords of other users are used as the fake indexes in honeyindex set, so it's difficult to find which password is wrong and which is correct becomes more complicated for an adversary.

5. CONCLUSIONS

In this research, we have analysed the security of the honeyword system and addressed a number offlaws that need to be managed before successful realization of the scheme. The strength of the honeyword system directly depends on the generation algorithm, i.e. flatness of the generator algorithm determines the chance of distinguishing the correct password out of respective sweetwords. Another thing that we would like to stress is that defined reaction policies in case of a honeywords entrance can be exploited by an adversary to realize a DoS attack. This will be a serious threat if the chance of an adversary in hitting a honeyword given the respective password is not negligible. We have noted that the security approach should strike a balance between DoS vulnerability and effectiveness of honeywords. As per the Imran Erguler, chaffing-by-tweaking method is weak and also some weaknesses of the chaffing-by-tweaking techniques are accepted by their creators. Moreover, the chaffing- with tough nuts model has been investigated, and we have

doubted about its favour as opposed to ideas of Juels and Rivest. Finally, we come up with a new approach to make the generation algorithm as close as to human nature by creating honeywords with randomly picking passwords that belong to other users in the system. In this we use SHA2 which becomes difficult to invert the hash values into the passwords as well as we shuffle the records in files and also the honeyindexes in the honeyindex set which makes guessing of correct password more complicated.

REFERENCES

[1] D. Mirante and C. Justin, "Understanding Password Database Compromises," Dept. of Computer Science and Engineering Polytechnic Inst. of NYU, Tech. Rep. TR-CSE-2013-02, 2013.

[2] A. Vance, "If Your Password is 123456, Just Make It Hackme,"The New York Times, vol. 20, 2010.

[3] K. Brown, "The Dangers of Weak Hashes," SANS Institute InfoSec Reading Room, Tech. Rep., 2013.

[4] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, "Password Cracking Using Probabilistic Context-Free Grammars," in Security and Privacy, 30th IEEE Symposium on. IEEE, 2009, pp. 391–405.

[5] F. Cohen, "The Use of Deception Techniques: Honeypots and Decoys," Handbook of Information Security, vol. 3, pp. 646–655, 2006.

[6] M. H. Almeshekah, E. H. Spafford, and M. J. Atallah, "Improving Security using Deception," Center for Education and Research Information Assurance and Security, Purdue University, Tech. Rep. CERIAS Tech Report 2013-13, 2013.

[7] C. Herley and D. Florencio, "Protecting financial institutions from brute-force attacks," in SEC'08, 2008, pp. 681–685.

[8] J. A and L. R. R, "Honeywords: Making Password cracking Detectable," in ACM SIGSAC conference on Computer & communications security, November 2013.

[9] Imran Erguler, "Achieving Flatness: Selecting the Honeywords from Existing User Passwords," IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 2, pp. 284 - 295, February 2015.

[10] Z. A. Genc, S. Kardas, and K. M. Sabir, "Examination of a New Defense Mechanism: Honeywords," Cryptology ePrint Archive,Report 2013/696, 2013.

[11] H. Bojinov, E. Bursztein, X. Boyen, and D. Boneh, "Kamouflage: Loss-resistant Password Management," in Computer Security– ESORICS 2010. Springer, 2010, pp. 286–302

[12] J. Bonneau, "The science of guessing: Analyzing an anonymized corpus of 70 million passwords," in Security and Privacy (SP), 2012IEEE Symposium on. IEEE, 2012, pp. 538–552.

[13] [21] D. Nagamalai, B. C. Dhinakaran, and J. K. Lee, "An In-depth Analysis of Spam and Spammers," arXiv preprint arXiv:1012.1665, 2010.